IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Utility Patent Application

VIRTUAL IMAGE GENERATION

Inventor(s):
Antonio Criminisi
Andrew Blake

ATTORNEY'S DOCKET NO. MS1-2067US
CLIENT'S DOCKET NO. 308594.01

EV436703231

VIRTUAL IMAGE GENERATION

Related Applications

[0001] This application is related to U.S. Patent No. 10/763,453, entitled "Virtual Camera Translation" and filed on January 23, 2004, which is incorporated herein by reference for all that it discloses and teaches.

[0002] This application is also related to U.S. Patent Application No. _____ [MSDocket No. 307820.01], entitled "Virtual Image Artifact Detection" and filed on April 16, 2004.

Technical Field

[0003] The invention relates generally to digital image processing, and more particularly to virtual image generation.

Background

[0004] Digital video cameras are useful in both consumer and professional contexts. Generally, digital video cameras capture sequences of digital images, which may then be transferred to a computer system for display or processing or to a storage device for storage. Alternatively, digital still cameras may be employed to capture distinct, non-video digital images.

[0005] In some applications, stereo cameras may be employed to generate synthetic virtual images from a variety of viewpoints. For example, in video conferencing application, a single video camera can capture images of a conference participant. However, the participant's gaze tends to align with a display window presented in a monitor by the video conferencing system, while the video camera is

typically mounted to the side of or above or below the display window, resulting in a misalignment between the participant's gaze and the captured video (e.g., capturing images of the side of the participant's head rather than a "straight-on" view). Accordingly, virtual images of straight-on, cyclopean views may be synthesized from stereo images captured by stereo cameras that are offset relative to the display window. It should be understood that other applications of virtual image generation may be employed outside the field of video conferencing.

[0006] However, existing methods of synthesizing cyclopean virtual images from stereo images often produce artifacts (e.g., streaks, blocks, and halos) that diminish the quality of the resulting virtual images. For example, an exemplary type of artifact results from mismatched mixes of foreground and background colors in corresponding pixels of stereo images. Because of the pixel disparities introduced by the relative positions and angles of the stereo cameras, it is common that the background colors contributing to the color mixing in one pixel of one stereo image are different from the background colors contributing to the color mixing in a corresponding pixel of another stereo image. A result of this color mixing difference is a tendency to inaccurately synthesize a virtual image pixel where color mixing discrepancies exists, thereby introducing incorrect transparency effects, streaking, and aliasing into the resulting virtual image.

Summary

[0007] Implementations described and claimed herein address the foregoing problems by detecting the artifacts in a virtual image generated from stereo images. A disparity map is generated from the stereo images, and individual projected images are

determined based on the disparity map and the corresponding stereo images. A difference map is then computed between the individual projected images to indicate the artifacts.

[0008] Having detected an artifact in the virtual image, a source patch in the virtual image is defined relative to the artifact. A target patch is generated using a split-patch search technique as a composite of a background exemplar patch and a foreground exemplar patch. Each exemplar patch may be identified from an image patch selected from at least one of the input stereo images. The source patch of the virtual image is replaced with the target patch to correct the detected artifact.

[0009] In some implementations, articles of manufacture are provided as computer program products. One implementation of a computer program product provides a computer program storage medium readable by a computer system and encoding a computer program. Another implementation of a computer program product may be provided in a computer data signal embodied in a carrier wave by a computing system and encoding the computer program.

[0010] The computer program product encodes a computer program for a computer process executing on a computer system. The computer process corrects an artifact in a virtual image synthesized from stereo images. An artifact is detected in the virtual image based on a disparity map of the stereo images. A source patch is designated relative to the artifact in the virtual image. A target patch is generated as a composite of a background exemplar patch and a foreground exemplar patch, each exemplar patch being identified from an image patch from at least one of the stereo images. The source patch of the virtual image is replaced with the target patch.

[0011] In another implementation, a method is provided that corrects an artifact in a virtual image synthesized from stereo images. An artifact is detected in the virtual image based on a disparity map of the stereo images. A source patch is designated relative to the artifact in the virtual image. A target patch is generated as a composite of a background exemplar patch and a foreground exemplar patch, each exemplar patch being identified from an image patch from at least one of the stereo images. The source patch of the virtual image is replaced with the target patch.

[0012] In another implementation, a system for correcting an artifact in a cyclopean virtual image synthesized from stereo images is provided. An artifact correction module detects the artifact in the virtual image based on a disparity map of the stereo images. The module designates a source patch relative to the artifact in the virtual image. The module also generates a target patch as a composite of a background exemplar patch and a foreground exemplar patch and replaces the source patch of the virtual image with the target patch.

[0013] Other implementations are also described and recited herein.

Brief Descriptions of the Drawings

[0014] FIG. 1 illustrates an exemplary artifact-correcting virtual image generation system.

[0015] FIG. 2 illustrates an exemplary video conferencing system configuration for generating an artifact-corrected virtual image.

[0016] FIG. 3 illustrates a cause of color mixing discrepancies in corresponding pixels of stereo images.

[0017] FIG. 4 illustrates a color mixing discrepancy in corresponding pixels of stereo images.

[0018] FIG. 5 depicts an exemplary system for removing artifacts from a virtual image.

[0019] FIG. 6 depicts operations in an exemplary artifact detection process.

[0020] FIG. 7 illustrates splitting a virtual image patch into foreground and background components based on a disparity map in an exemplary implementation.

[0021] FIG. 8 illustrates an exemplary split-patch search of candidate exemplar patches.

[0022] FIG. 9 illustrates results of an exemplary selection of candidate exemplar patches.

[0023] FIG. 10 illustrates exemplary occlusion of a background region of a patch.

[0024] FIG. 11 illustrates a system useful for implementing an embodiment of the present invention.

Detailed Description

[0025] FIG. 1 illustrates an exemplary artifact-correcting virtual image generation system 100. In the system 100, a left image 102 is captured by a camera mounted on the right side of the video display, as seen by the user. Likewise, a right image 104 is captured by a camera mounted on the left side of the video display, as seen by the user. As such, in both images, the user can be seen looking into the video display, as opposed to looking directly at one of the cameras. The left and right images 102 and 104 are input to a virtual image synthesis module 106, which

generates from the images 102 and 104 a virtual image 108 with gaze correction. The virtual image synthesis module 106 may also generate an occlusion map 110 and a stereo disparity map 112, as shown in FIG. 1. Exemplary methods of synthesizing virtual images from stereo images are described in previously-incorporated application U.S. Patent No. 10/763,453, entitled "Virtual Camera Translation".

[0026] The virtual image 108, the occlusion map 110, and the stereo disparity map 112 generated by the virtual image synthesis module 106 are input to an artifact correction module 114 to generate the virtual image 116 with gaze correction and artifact correction. As a result, the virtual image 116 depicts a high-quality image of the user appearing to look directly into the camera.

[0027] FIG. 2 illustrates an exemplary video conferencing system 200 configuration for generating an artifact-corrected virtual image. A computer system 202 is coupled to a video display 204 having two cameras 206 and 208 mounted on either side of the video display 204. It should be understood that other stereo placements of the cameras 206 and 208 (e.g., top/bottom, a four-camera configuration at each corner, etc.). A video window 210 displays a remote participant on the other end of the video conference session.

[0028] In a configuration having only a single camera, the user typically focuses his or her eyes on the video window 210, while the single camera captures images of the user from one side or the other. As such, the captured images sent to the remote participant are primarily a side view of the user's head, not the desired straight-on view of the user's face. The illustrated configuration, however, allows synthesis of a cyclopean virtual image from the captured left and right images of the

user. It should be understood that cyclopean refers to the single virtual image. Furthermore, in one implementation, the cyclopean virtual image may be displayed at different video window locations on the display screen (i.e., cyclopean virtual image location is not limited to a central orientation relative to the stereo cameras) while maintaining alignment of the virtual camera with the user's gaze. Likewise, axial translation of the virtual image may also be achieved in an implementation.

[0029] It should be understood that more than two cameras may also be used to generate a cyclopean virtual image. Likewise, the cameras may be in alternative orientations, such as at the top and bottom of the video display. For example, one configuration may include four cameras, each placed at a corner of the video display.

[0030] FIG. 3 illustrates a cause of color mixing discrepancies in corresponding pixels of stereo images. A foreground object 300 is shown against a multi-color background 302, where the solid line 304 represents one color A and the broken line 306 represents another color B, and both background colors differ from the color of the foreground object.

[0031] A right camera 308 captures a right image 310 that includes a pixel 312 having color of the foreground object, a pixel 314 having the color of the background B, and a pixel 316 having a color mix of the foreground object and the background B. A left camera 318 captures a left image 320 that includes a pixel 322 having color of the foreground object, a pixel 324 having the color of the background A, and a pixel 326 having a color mix of the foreground object and the background A. As discussed, the different color mixes of the corresponding pixels 316 and 326 may produce artifacts caused by mismatching the pixels during the virtual image synthesis

process (e.g., generation of the disparity map is inaccurate because pixel 316 is not matched with pixel 326 because of the color mixing difference).

[0032] FIG. 4 illustrates a color mixing discrepancy in corresponding pixels 400 and 402 of stereo images 404 (left) and 406 (right). An image 408 represents a magnified version of region 410 of left image 404, and an image 412 represents a magnified version of region 414 of right image 406. As is evident from the magnified images 408 and 412, the corresponding pixels 400 and 402 include colors of the subject's shoulder and colors of the background. However, a dark brown door provides the background color contribution in the pixel 400 while a light tan wall provides the background color contribution in the pixel 402. As a result, the colors of the corresponding pixels 400 and 402 are different.

[0033] When generating a virtual image pixel corresponding to the pixels 400 and 402, the color mixing discrepancy between the two pixels can result in a mismatch as a disparity graph is generated along the epipolar lines associated with the two pixels. Color mixing mismatches typically result in artifacts in the virtual image (e.g., artifacts 500 in FIG. 5). However, such artifacts can be detected and corrected (e.g., reduced) using techniques described herein.

[0034] FIG. 5 depicts an exemplary system for removing artifacts 500 from a virtual image 502. The artifacts 500 result from color mixing mismatches, such as the mismatch illustrated in FIG. 4. An artifact correction module 504 corrects the artifacts to provide a more suitable result image 506, in which the corresponding corrected region 508 shows dramatically reduced artifact effects. Other artifact regions in the images also show improved results (see regions 510 and 512).

[0035] FIG. 6 depicts operations in an exemplary artifact detection process. In the illustrated implementation, a left image 600 (denoted as I_l) and a right image 602 (denoted as I_r) are processed (e.g., "epipolar-rectified") by a disparity map generator 604 to generate from the two images a stereo disparity map 606 (denoted as *D*) and an occlusion map 607 (denoted as *O*). A couple of techniques for generating cyclopean virtual images from stereo images are described in previously incorporated U.S. Patent No. 10/763,453, entitled "Virtual Camera Translation", although other techniques may be used. The disparity map *D* is generated with respect to the coordinate system defined by the desired virtual viewpoint (e.g., the desired location of the virtual camera, such as the display window of a video conferencing system). A rough virtual image *I* (not shown) also generated from images I_l and I_r by techniques described in the previously incorporated U.S. Patent No. 10/763,453, entitled "Virtual Camera Translation" or any other known or equivalent method. The rough virtual image *I* includes artifacts, such those artifacts introduced by color mixing mismatches.

[0036] The left image 600, the disparity map 606, and the occlusion map 607 are input to a disparity-driven image warp module 608 to generate a projected left image 610 with occlusion regions (i.e., the solid light grey and solid dark grey regions to the left and right of the subject). Likewise, the right image 602, the disparity map 606, and the occlusion map 607 are input to a disparity-driven image warp module 612 to generate a projected right image 614 with occlusion regions. It should be understood that modules 608 and 612 are illustrated as distinct modules but could be implemented as the same module taking different inputs (e.g., left and right images). The projected images 610 and 614 (denoted as I_l^w for the projected left

image and I_r^w for the right projected right image) represent projections of the corresponding (e.g., left or right) images into a target viewpoint. The light grey and dark grey regions represent left and right half-occlusions.

[0037] A pixel-wise color distance $d\left(I_l^w, I_r^w\right)$ between the two projected images is computed by an aliasing-insensitive image distancing module 616 to indicate the location and entity of artifacts, as shown in the difference map 618. In one implementation, artifacts are represented as:

the set $\mathcal{A}$ *of pixels* $\mathbf{p} \in I$ *such that* $d\left(I_l^w, I_r^w\right) > \lambda$ *(e.g.,* $\lambda=5$*).*

[0038] Therefore, the image noise of the difference map 618 may be filtered (e.g., $d\left(I_l^w, I_r^w\right) > \lambda$), for example, by a thresholding module 620, to generate an artifact map 622. Assuming low levels of image noise, large values of $d\left(I_l^w, I_r^w\right)$ in the artifact map 622 occur in pixel locations where the virtual image synthesis algorithm has failed to correctly estimate the correct pixel correspondence between the two images I_l and I_r (e.g., because of color mixing mismatches or other matching errors).

[0039] FIG. 7 illustrates splitting a virtual image patch into foreground and background filter components based on a disparity map in an exemplary implementation. As discussed, the artifact map 622 of FIG. 6 indicates pixel locations of artifacts in the rough virtual image generated by a virtual image synthesis module. By virtue of the virtual image synthesis, a rough virtual image I, a disparity map D, and a set $\mathcal{A}$ of artifacts are available as inputs to an artifact correction operation.

[0040] For each pixel $\mathbf{p} \in \mathcal{A}$, a source patch $\Phi_\mathbf{p}$ centered at $\mathbf{p}$ is defined. For example, a source patch may include a set of neighboring pixels, such as a square region that is 5 pixels wide and 5 pixels high and centered at $\mathbf{p}$. In the illustration, larger patches are shown to assist in the description. An artifact correction operation searches for a new target patch $\Psi_\mathbf{p}$ with which to replace the source patch $\Phi_\mathbf{p}$. The new target patch $\Psi_\mathbf{p}$ will be similar to the source patch $\Phi_\mathbf{p}$ but with the artifacts removed. Replacing a source patch $\Phi_\mathbf{p}$ with a new target patch $\Psi_\mathbf{p}$ for all pixels $\mathbf{p}$ removes detected artifacts throughout the entire image.

[0041] In FIG. 7, an artifact point 700 (denoted as $\mathbf{p}$) in a rough virtual image 702 (denoted as I) is selected. A source patch 704 (denoted as $\Phi_\mathbf{p}$) relative to the artifact 700 is defined in the rough virtual image 702. A disparity point 706 (denoted again as $\mathbf{p}$, because its location corresponds to the location of point 700) in a disparity map 708 (denoted as D) is identified as corresponding to the pixel $\mathbf{p}$ in the rough virtual image 702. A corresponding disparity patch 710 (denoted as $D_\mathbf{p}$) is also identified. The disparity patch 710 is smoothed by a filter module 712 to provide a filtered disparity patch $\tilde{D}_\mathbf{p}$ (not shown). By filtering the patch (e.g., by performing a low-pass smoothing operation on the patch), high frequency components of the disparity signal, which often arise from matching mistakes, are removed. Given these inputs, a foreground weight array $\Omega_\mathbf{p}^f$ and a background weight array $\Omega_\mathbf{p}^b$ may be computed by a weighting module 714 as follows:

$$\Omega_\mathbf{p}^f = \frac{\tilde{D}(\mathbf{q}) - \tilde{D}^m}{\tilde{D}^M - \tilde{D}^m};\ \Omega_\mathbf{p}^b = 1 - \Omega_\mathbf{p}^f;\ \forall \mathbf{q} \in \Phi_\mathbf{p}$$

with $\tilde{D}^m$ and $\tilde{D}^M$ representing the minimum and maximum values respectively of the disparities within the filtered disparity patch $\tilde{D}_\mathbf{p}$. A map 716 representing a foreground weighting array, and a map 718 representing a background weighting array are shown as outputs from the weighting module 714. The weighting arrays may also be referred to as "filter maps".

[0042] FIG. 8 illustrates an exemplary split-patch search of candidate exemplar patches. The split-patch search searches for patches in the original right and left images that are most similar to the foreground and background portions of the source patch from the virtual image. In one implementation, the search is performed along the scan line (e.g., an epipolar line) corresponding to point **p** up to a distance δy in each of the original images. An exemplary value of δy may be computed as follows:

$$\delta y = \max_{\mathbf{q} \in I | \mathbf{q}_y = y} \frac{\tilde{D}(\mathbf{q})}{2}$$

[0043] An artifact and a corresponding source patch 800 (shown in exploded view 802 and denoted as $\Phi_\mathbf{p}$) are selected from a rough virtual image 804. In the illustrated implementation, the artifact is detected by operations described with regard to FIG. 6, although other artifact detection methods may be employed. A foreground filter 806 (denoted as $\Omega_\mathbf{p}^f$) is used in combination with the source patch 802 and generic left/right-view patches to compute candidate foreground exemplar patches $R_\mathbf{p}^f$ and $L_\mathbf{p}^f$. Likewise, background filter 806 (denoted as $\Omega_\mathbf{p}^b$) is used in combination with the source patch 802 and generic left/right-view patches to compute candidate background exemplar patches $R_\mathbf{p}^b$ and $L_\mathbf{p}^b$.

[0044] Choosing the right image 810 to exemplify split-patch search operations, generic patches along the epipolar line that includes the pixel **p** in the right image are evaluated against the source patch, as modified by the foreground and background filters (see the equations below), to identify the candidate exemplar patches. Selecting the minimum difference between the pixel-wise parameters below (e.g., $\Omega_{\mathbf{p}}^{f} * \Phi_{\mathbf{p}}$ and $\Omega_{\mathbf{p}}^{f} * R_{\mathbf{q}}$) represents one exemplary implementation of such a split-patch search. The graphs 812 and 814 illustrate the evaluations used to select the candidate exemplar patches 816 (foreground) and 818 (background) from the right image 810. The process is repeated for the left image (not shown)

[0045] Algorithmically, given appropriate search constraints, candidate exemplar patches may be determined as follows:

$$R_{\mathbf{p}}^{f} = \arg \min_{p_x - \delta_y \le q_x \le p_x} d(\Omega_{\mathbf{p}}^{f} * \Phi_{\mathbf{p}}, \Omega_{\mathbf{p}}^{f} * R_{\mathbf{q}})$$

$$R_{\mathbf{p}}^{b} = \arg \min_{p_x - \delta_y \le q_x \le p_x} d(\Omega_{\mathbf{p}}^{b} * \Phi_{\mathbf{p}}, \Omega_{\mathbf{p}}^{b} * R_{\mathbf{q}})$$

$$L_{\mathbf{p}}^{f} = \arg \min_{p_x \le q_x \le p_x + \delta_y} d(\Omega_{\mathbf{p}}^{f} * \Phi_{\mathbf{p}}, \Omega_{\mathbf{p}}^{f} * L_{\mathbf{q}})$$

$$L_{\mathbf{p}}^{b} = \arg \min_{p_x \le q_x \le p_x + \delta_y} d(\Omega_{\mathbf{p}}^{b} * \Phi_{\mathbf{p}}, \Omega_{\mathbf{p}}^{b} * L_{\mathbf{q}})$$

with $L_{\mathbf{q}}$ and $R_{\mathbf{q}}$ representing the generic left and right view patches centered at the generic point $\mathbf{q}|q_y=p_y$ along the epipolar line. The symbol "*" represents point-wise multiplication between images (or patches). In one implementation, the distance $d(\Pi_1,\Pi_2)$ between two generic patches Π_1 and Π_2 is represented as the sum of squared differences (SSD) of pixel values, where the pixels of $\mathcal{A}$ are ignored.

[0046] FIG. 9 illustrates results of an exemplary selection of candidate exemplar patches. A rough virtual image 900, a right image 902, and a left image 904 are processed in a manner described with regard to FIG. 8, resulting in a source patch 906, a right candidate foreground exemplar patch 908, a right candidate background exemplar patch 910, a left candidate foreground exemplar patch 912, and a left candidate background exemplar patch 914. These resulting candidate exemplar patches are considered "candidates" because occlusion can render one of the background candidate exemplar patches meaningless. Indeed, the true background patch of $\Phi_{\mathbf{p}}$ has been occluded in the left view, thus the retrieved patch $L_{\mathbf{p}}^{b}$ is meaningless. In contrast, the right candidate background exemplar patch $R_{\mathbf{p}}^{b}$ contains the correct background information.

[0047] Determining an uncontaminated background exemplar patch $\Pi_{\mathbf{p}}^{b}$ may be performed automatically by selecting the background patch that is most similar to the background of the source patch 906, using:

$$\Pi_{\mathbf{p}}^{b} = \arg\min_{\Lambda \in \{L,R\}} d\left(\Omega_{\mathbf{p}}^{b} * \Lambda_{\mathbf{p}}^{b}, \Omega_{\mathbf{p}}^{b} * \Phi_{\mathbf{p}}\right)$$

[0048] The uncontaminated background exemplar patch $\Pi_{\mathbf{p}}^{b}$ is one component used to generate a new composite target patch $\Psi_{\mathbf{p}}$. In addition, candidate exemplar foreground patches $R_{\mathbf{p}}^{f}$ and $L_{\mathbf{p}}^{f}$ have been already determined for each pixel $\mathbf{p} \in \mathcal{A}$. An effective algorithm for compositing a target patch $\Psi_{\mathbf{p}}$ may be stated as:

$$\Psi_{\mathbf{p}} = \Gamma_{\mathbf{p}} * \Pi_{\mathbf{p}}^{f} + (1 - \Gamma_{\mathbf{p}}) * \Pi_{\mathbf{p}}^{b} \qquad (1)$$

where $\Gamma_{\mathbf{p}}$ represents the transparency of the pixel **p**, $\Pi_{\mathbf{p}}^{f}$ represents the uncontaminated foreground exemplar patch about the pixel **p**, and $\Pi_{\mathbf{p}}^{b}$ represents the uncontaminated background exemplar patch about the pixel **p**. However, $\Pi_{\mathbf{p}}^{f}$ has not yet been computed and, in fact, an exact solution for $\Pi_{\mathbf{p}}^{f}$ does not appear available. As such, reasonable approximations are employed in one implementation.

[0049] The candidate exemplar foreground patch $L_{\mathbf{p}}^{f}$ may be interpreted itself as a composite image. Its background (i.e., the poster on the back wall) is completely visible in the right input view I_r. The background of $L_{\mathbf{p}}^{f}$ can be extracted by the following search process:

$$\hat{L}_{\mathbf{p}}^{b} = \arg \min_{p_x \le q_x \le p_x + \delta y} d\left(\Omega_{\mathbf{p}}^{b} * L_{\mathbf{p}}^{f}, \Omega_{\mathbf{p}}^{b} * R_{\mathbf{q}}\right)$$

[0050] The analogous background $\hat{R}_{\mathbf{p}}^{b}$ corresponding to the right foreground patch $R_{\mathbf{p}}$ is occluded by the subject's head and therefore cannot be copied directly from either of the two input views. For example, in FIG. 10, the background of the right patch $R_{\mathbf{p}}^{f}$ 1000 centered at point **p** (i.e., point 1002) is, of course, occluded in part by the foreground of the right image I_r 1004. Furthermore, the background of the right patch $R_{\mathbf{p}}^{f}$ 1000 cannot be found along the scan line 1006 of the left image 1008. Accordingly, an approximation of $\hat{R}_{\mathbf{p}}^{b}$ is computed by one of a variety of methods.

[0051] An exemplary method of approximating $\hat{R}_{\mathbf{p}}^{b}$ may be implemented as follows. Given the right foreground patch $R_{\mathbf{p}}^{f}$ and the background filter $\Omega_{\mathbf{p}}^{b}$, pixels of $R_{\mathbf{p}}^{f}$ that belong to the background (i.e., the door in the example image) are extracted

and a parametric surface model is fit (e.g., polynomial, spline, etc.) to the corresponding color values in RGB space. Then, the fitted surface model is used to extrapolate the colors of the pixels in the occluded portion of R_p^f (i.e., behind the hair in the example). In one implementation applied to small patches (e.g., 5 pixels by 5 pixels), extrapolation via a generic planar fit (generally not constant) has produced acceptable results. Symmetrical reasoning is applied when $\hat{L}_p^b$ is occluded.

[0052] Based on the approximation operation, two foreground patches (L_p^f and R_p^f) and two corresponding background patches ($\hat{L}_p^b$ and $\hat{R}_p^b$) have been extracted. Therefore, the conventional compositing equation may be stated for each candidate foreground exemplar patch:

$$L_p^f = \Gamma_p * \Pi_p^f + \left(1 - \Gamma_p\right) * \hat{L}_p^b \tag{2}$$

$$R_p^f = \Gamma_p * \Pi_p^f + \left(1 - \Gamma_p\right) * \hat{R}_p^b \tag{3}$$

with Γ_p representing the transparencies and Π_p^f representing the uncontaminated foreground colors. Given that both background patches ($\hat{L}_p^b$ and $\hat{R}_p^b$) are known, then both Γ_p and Π_p^f may now be uniquely determined from Equations (2) and (3). Transparencies may be assumed to apply equally to each of the RGB channels.

[0053] Noise and coincidently similar colors shared by corresponding pixels can corrupt the accurate recovery of transparencies and foreground colors. However, such recovery can be improved through incorporation of prior information (e.g., on the distribution of alpha (i.e., transparency) and color value). In one implementation, a Bayesian approach may be applied to regularize the alpha and color data.

Alternative approaches are also available, including filtering the extracted alpha and color data in a depth-dependent fashion.

[0054] As a result, given the foreground exemplar patch Π_p^f, the transparency Γ_p, and the background Π_p^b, the target patch may be computed according to Equation (1).

[0055] The exemplary hardware and operating environment of FIG. 11 for implementing the invention includes a general purpose computing device in the form of a computer 20, including a processing unit 21, a system memory 22, and a system bus 23 that operatively couples various system components include the system memory to the processing unit 21. There may be only one or there may be more than one processing unit 21, such that the processor of computer 20 comprises a single central-processing unit (CPU), or a plurality of processing units, commonly referred to as a parallel processing environment. The computer 20 may be a conventional computer, a distributed computer, or any other type of computer; the invention is not so limited.

[0056] The system bus 23 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, a switched fabric, point-to-point connections, and a local bus using any of a variety of bus architectures. The system memory may also be referred to as simply the memory, and includes read only memory (ROM) 24 and random access memory (RAM) 25. A basic input/output system (BIOS) 26, containing the basic routines that help to transfer information between elements within the computer 20, such as during start-up, is stored in ROM 24. The computer 20 further includes a hard disk drive 27 for reading from and

writing to a hard disk, not shown, a magnetic disk drive 28 for reading from or writing to a removable magnetic disk 29, and an optical disk drive 30 for reading from or writing to a removable optical disk 31 such as a CD ROM or other optical media.

[0057] The hard disk drive 27, magnetic disk drive 28, and optical disk drive 30 are connected to the system bus 23 by a hard disk drive interface 32, a magnetic disk drive interface 33, and an optical disk drive interface 34, respectively. The drives and their associated computer-readable media provide nonvolatile storage of computer-readable instructions, data structures, program modules and other data for the computer 20. It should be appreciated by those skilled in the art that any type of computer-readable media which can store data that is accessible by a computer, such as magnetic cassettes, flash memory cards, digital video disks, random access memories (RAMs), read only memories (ROMs), and the like, may be used in the exemplary operating environment.

[0058] A number of program modules may be stored on the hard disk, magnetic disk 29, optical disk 31, ROM 24, or RAM 25, including an operating system 35, one or more application programs 36, other program modules 37, and program data 38. A user may enter commands and information into the personal computer 20 through input devices such as a keyboard 40 and pointing device 42. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 21 through a serial port interface 46 that is coupled to the system bus, but may be connected by other interfaces, such as a parallel port, game port, or a universal serial bus (USB). A monitor 47 or other type of display device is

also connected to the system bus 23 via an interface, such as a video adapter 48. In addition to the monitor, computers typically include other peripheral output devices (not shown), such as speakers and printers.

[0059] The computer 20 may operate in a networked environment using logical connections to one or more remote computers, such as remote computer 49. These logical connections are achieved by a communication device coupled to or a part of the computer 20; the invention is not limited to a particular type of communications device. The remote computer 49 may be another computer, a server, a router, a network PC, a client, a peer device or other common network node, and typically includes many or all of the elements described above relative to the computer 20, although only a memory storage device 50 has been illustrated in FIG. 11. The logical connections depicted in FIG. 11 include a local-area network (LAN) 51 and a wide-area network (WAN) 52. Such networking environments are commonplace in office networks, enterprise-wide computer networks, intranets and the Internet, which are all types of networks.

[0060] When used in a LAN-networking environment, the computer 20 is connected to the local network 51 through a network interface or adapter 53, which is one type of communications device. When used in a WAN-networking environment, the computer 20 typically includes a modem 54, a network adapter, a type of communications device, or any other type of communications device for establishing communications over the wide area network 52. The modem 54, which may be internal or external, is connected to the system bus 23 via the serial port interface 46. In a networked environment, program modules depicted relative to the personal

computer 20, or portions thereof, may be stored in the remote memory storage device. It is appreciated that the network connections shown are exemplary and other means of and communications devices for establishing a communications link between the computers may be used.

[0061] In an exemplary implementation, a virtual image synthesis module, an artifact removal module, an image distancing module, a disparity/occlusion map generator, and other modules may be incorporated as part of the operating system 35, application programs 36, or other program modules 37. Virtual image data, image data, image color distances, map data, and other data may be stored as program data 38.

[0062] The embodiments of the invention described herein are implemented as logical steps in one or more computer systems. The logical operations of the present invention are implemented (1) as a sequence of processor-implemented steps executing in one or more computer systems and (2) as interconnected machine modules within one or more computer systems. The implementation is a matter of choice, dependent on the performance requirements of the computer system implementing the invention. Accordingly, the logical operations making up the embodiments of the invention described herein are referred to variously as operations, steps, objects, or modules.

[0063] The above specification, examples and data provide a complete description of the structure and use of exemplary embodiments of the invention. Since many embodiments of the invention can be made without departing from the

spirit and scope of the invention, the invention resides in the claims hereinafter appended.